Question Bank ET & IoT

Q.1 What is break and continue statement?

Ans :- The break statement terminates a `while` or `for` loop completely. The continue statement terminates execution of the statements within a `while` or `for` loop and continues the loop in the next iteration.

Q.2 Discuss the features of Keil Compiler.

Ans:- The Keil C51 C Compiler for the 8051 microcontroller is the most popular 8051 C compiler in the world. It provides more features than any other 8051 C compiler available today.

The C51 Compiler allows you to write 8051 microcontroller applications in C that, once compiled, have the efficiency and speed of assembly language. Language extensions in the C51 Compiler give you full access to all resources of the 8051.

The C51 Compiler translates C source files into relocatable object modules which contain full symbolic information for debugging with the µVision Debugger or an in-circuit emulator. In addition to the object file, the compiler generates a listing file which may optionally include symbol table and cross reference information.

Q.3 Discuss Arduino Uno board and also mention the number of digital pin and analog pin in it.

Ans:- As we discussed we know that Arduino Uno is the most standard board available and probably the best choice for a beginner. We can directly connect the board to the computer via a USB Cable which performs the function of supplying the power as well as acting as a serial port.

Vin: This is the input voltage pin of the Arduino board used to provide input supply from an external power source.

5V: This pin of the Arduino board is used as a regulated power supply voltage and it is used to give supply to the board as well as onboard components.

3.3V: This pin of the board is used to provide a supply of 3.3V which is generated from a voltage regulator on the board

GND: This pin of the board is used to ground the Arduino board.

Reset: This pin of the board is used to reset the microcontroller. It is used to Resets the microcontroller.

Analog Pins: The pins A0 to A5 are used as an analog input and it is in the range of 0-5V.

Digital Pins: The pins 0 to 13 are used as a digital input or output for the Arduino board.

Serial Pins: These pins are also known as a UART pin. It is used for communication between the Arduino board and a computer or other devices. The transmitter pin number 1 and receiver pin number 0 is used to transmit and receive the data resp

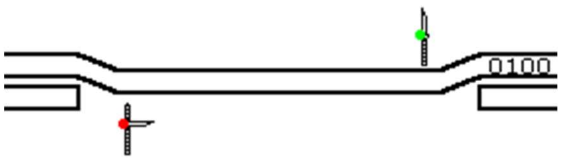
Q.4 Discuss control statements used in programming language.
Ams:- The control statements used in the C language help a user to specify a program control's flow. In simpler words, the control statements help users specify the order of execution of the instructions present in a program. These make it possible for the program to make certain decisions, perform various tasks repeatedly, or even jump from any one section of the code to a different section.
In C language, the control of the program flows from a given instruction to another. This type of control flow that occurs from any given command to another is known as the sequential control flow. Now, in any C program, a programmer might want to repeat some sets of instructions or even skip the instructions when they are writing logic. Declarations in C, also known as control declarations or decision-making, help them in making such decisions.

Q.5 What is serial communication? Discuss the different transmission modes in serial communication.
:-Serial Communication Data Transmission Modes means transferring of data between two devices. It is also called communication mode. These modes direct the direction of flow of information. There are three types of transmission mode.

They are :

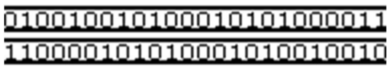
- Simplex Mode
- Half duplex Mode
- Full duplex Mode

Data in a simplex channel is always one way. Simplex channels are not often used because it is not possible to send back error or control signals to the transmit end.
Half-Duplex



A half-duplex channel can send and receive, but not at the same time. It's like a one-lane bridge where two way traffic must give way in order to cross. Only one end transmits at a time, the other end receives. In addition, it is possible to perform error detection and request the sender to retransmit information that arrived corrupted.

Full-Duplex



Data can travel in both directions simultaneously. There is no need to switch from transmit to receive mode like in half duplex.

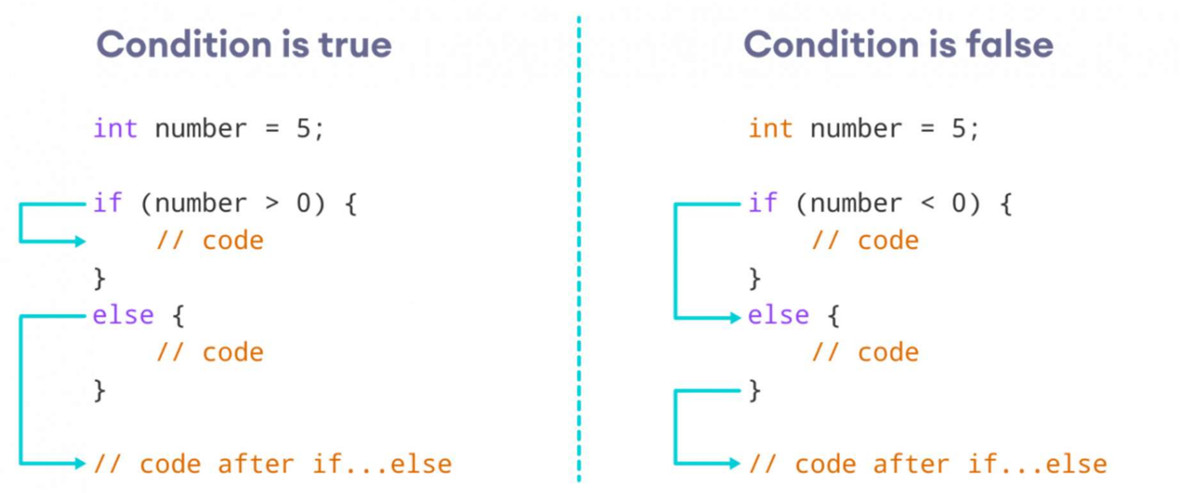
Full-Duplex is like the ordinary two-lane highway. In some cases, where traffic is heavy enough, a railroad will decide to lay a double track to allow trains to pass in both directions. In communications, this is most common with networking. Our fiber optic hubs have two connectors on each port, one for each lane of a two-lane roadway. Full-Duplex fiber is two cables bundled or tied together to form the two-lane roadway.

Q.6 What is the Syntax of if , if-else and nested if-else?

The `if` statement evaluates the `condition` inside the parentheses `( )`.

- If the `condition` evaluates to `true`, the code inside the body of `if` is executed.
- If the `condition` evaluates to `false`, the code inside the body of `if` is skipped.

The `if..else` statement evaluates the `condition` inside the parenthesis.



```
int number = 5;

if (number > 0) {
    // code
}
else {
    // code
}

// code after if...else
```

```
int number = 5;

if (number < 0) {
    // code
}
else {
    // code
}

// code after if...else
```

How if...else Statement Works

If the `condition` evaluates `true`,

- the code inside the body of `if` is executed
- the code inside the body of `else` is skipped from execution

If the `condition` evaluates `false`,

- the code inside the body of `else` is executed
- the code inside the body of `if` is skipped from execution

The `if...else` statement is used to execute a block of code among two alternatives. However, if we need to make a choice between more than two alternatives, we use the `if...else if...else` statement.

The syntax of the `if...else if...else` statement is:

```
if (condition1) {
  // code block 1
}
else if (condition2){
  // code block 2
}
else {
  // code block 3
}
```

Here,

- If `condition1` evaluates to `true`, the `code block 1` is executed.
- If `condition1` evaluates to `false`, then `condition2` is evaluated.

- If `condition2` is `true`, the `code block 2` is executed.
- If `condition2` is `false`, the `code block 3` is executed.

Q.7 What is a sensor? Discuss the different types of sensors and their applications.
There are numerous definitions as to what a sensor is but I would like to define a Sensor as an input device which provides an output (signal) with respect to a specific physical quantity (input).
It is a device that converts signals from one energy domain to electrical domain. The definition of the Sensor can be better understood if we take an example in to consideration.

1. Temperature Sensor
2. Proximity Sensor
3. Accelerometer
4. IR Sensor (Infrared Sensor)
5. Pressure Sensor
6. Light Sensor
7. Ultrasonic Sensor
8. Smoke, Gas and Alcohol Sensor
9. Touch Sensor
10. Color Sensor
11. Humidity Sensor
12. Position Sensor
13. Magnetic Sensor (Hall Effect Sensor)
14. Microphone (Sound Sensor)
15. Tilt Sensor
16. Flow and Level Sensor
17. PIR Sensor
18. Touch Sensor
19. Strain and Weight Sensor

Q.8 Write a program to make switch operated LED light and draw the proteus diagram for the same.
https://roboticsbackend.com/wp-content/uploads/2020/12/arduino_led_push_button-1024x505.png

```
#define LED_PIN 8
#define BUTTON_PIN 7
void setup() {
pinMode(LED_PIN, OUTPUT);
pinMode(BUTTON_PIN, INPUT);
}
void loop() {
if (digitalRead(BUTTON_PIN) == HIGH) {
digitalWrite(LED_PIN, HIGH);
}
else {
digitalWrite(LED_PIN, LOW);
}
}
```

Q.9 Compare while loop and a do while loop.

Here, the main difference between a while loop and do while loop is that while loop check condition before iteration of the loop. On the other hand, the do-while loop verifies the condition after the execution of the statements inside the loop.

Difference Between while and do-while Loop (with Comparison Chart) - Tech Differences

Q.10 Differentiate between Two Way Selection and Multiway Selection with proper syntax, flowchart and suitable example.

What if you want to pick between two possibilities? If you are trying to decide between a couple of things to do, you might flip a coin and do one thing if it lands as heads and another if it is tails. In programming, you can use the **if** keyword followed by a statement or block of statements and then the **else** keyword also followed by a statement or block of statements.

```
// A block if/else statement
if (boolean expression)
{
   statement1;
   statement2;
}
else
{
   do other statement;
   and another one;
}
// A single if/else statement
if (boolean expression)
   Do statement;
else
   Do other statement;
```

The following flowchart demonstrates that if the condition (the boolean expression) is true, one block of statements is executed, but if the condition is false, a different block of statements inside the else clause is executed.

https://runestone.academy/ns/books/published/csawesome/_images/Condition-two.png

- A multi-way selection statement is used to execute **at most ONE** of the choices of a set of statements presented.
- **Syntax** of the multi-way select statement:

```
switch ( EXPRESSION )
{
   case CONSTANT1: one or more statements; break;

   case CONSTANT2: one or more statements; break;
   ...

   [default: one or more statements;]
}
```

- **Example:**

```
switch ( Avg )
{
   case 90:
   case 91:
   ...
   case 100: printf("Grade is A"); break;

   case 80:
   case 81:
   ...
   case 89: printf("Grade is B"); break;

   case 70:
   case 71:
   ...
   case 79: printf("Grade is C"); break;

   case 60:
   case 61:
   ...
   case 69: printf("Grade is D"); break;

   default: printf("Grade is F");
}
```

Q.11 Discuss the basic commands and Functions for Arduino programming with proper syntax.

Syntax in Arduino signifies the rules need to be followed for the successful uploading of the Arduino program to the board. The syntax of Arduino is **similar to the grammar in English**. It means that the rules must be followed in order to compile and run our code successfully.

For controlling the Arduino board and performing computations.

Digital I/O

digitalRead()
digitalWrite()
pinMode()

Analog I/O

analogRead()
analogReference()
analogWrite()

Zero, Due & MKR Family

analogReadResolution()
analogWriteResolution()

Advanced I/O

noTone()
pulseIn()
pulseInLong()
shiftIn()
shiftOut()
tone()

Time

delay()
delayMicroseconds()
micros()
millis()

Math

abs()
constrain()
map()
max()
min()
pow()
sq()
sqrt()

Trigonometry

cos()
sin()
tan()

Characters

isAlpha()
isAlphaNumeric()
isAscii()
isControl()
isDigit()
isGraph()

isHexadecimalDigit()
isLowerCase()
isPrintable()
isPunct()
isSpace()
isUpperCase()
isWhitespace()

Random Numbers

random()
randomSeed()

Bits and Bytes

bit()
bitClear()
bitRead()
bitSet()
bitWrite()
highByte()
lowByte()

External Interrupts

attachInterrupt()
detachInterrupt()

Interrupts

interrupts()
noInterrupts()

Communication

Serial
SPI
Stream
Wire

USB

Keyboard
Mouse

Variables

Arduino data types and constants.

Constants

HIGH | LOW
INPUT | OUTPUT | INPUT_PULLUP
LED_BUILTIN
true | false
Floating Point Constants
Integer Constants

Conversion

(unsigned int)
(unsigned long)
byte()
char()
float()
int()
long()
word()

Data Types

array
bool
boolean
byte
char
double
float
int
long
short
size_t
string
String()
unsigned char
unsigned int
unsigned long
void
word

Variable Scope & Qualifiers

const
scope

static
volatile

Utilities

PROGMEM
sizeof()

Structure

The elements of Arduino (C++) code.

Sketch

loop()
setup()

Control Structure

break
continue
do...while
else
for
goto
if
return
switch...case
while

Further Syntax

#define (define)
#include (include)
/* */ (block comment)
// (single line comment)
; (semicolon)
{} (curly braces)

Arithmetic Operators

% (remainder)
* (multiplication)
+ (addition)
- (subtraction)

Back to top

Q.12 Write an Arduino code to glow one LED ON for 0.1 sec and OFF for 100 miliseconds.

```
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(100);              // wait for a second
  digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
  delay(100);              // wait for a second
}
```

Q.13 With an example sketch, explain multidimensional array.
A multi-dimensional array can be termed as an array of arrays that stores homogeneous data in tabular form. Data in multidimensional arrays are stored in row-major order.
https://i.stack.imgur.com/nX0uo.jpg

Q.14 Develop the coding for arduino interfacing with LCD 2x16.
Interface 16x2 LCD (parallel interface) with Arduino Uno - Arduino Project Hub

Q.15 Write a program to make switch operated blinking of one LED ON for 2 sec and OFF for 1 sec.
Que no. 8

Q.16 Eloborate the following functions: (i) analogRead() and (ii) delayMicroseconds()

Pauses the program for the amount of time (in microseconds) specified by the parameter. There are a thousand microseconds in a millisecond and a million microseconds in a second.

Currently, the largest value that will produce an accurate delay is 16383; larger values can produce an extremely short delay. This could change in future Arduino releases. For delays longer than a few thousand microseconds, you should use `delay()` instead.

Reads the value from the specified analog pin. Arduino boards contain a multichannel, 10-bit analog to digital converter. This means that it will map input voltages between 0 and the operating voltage(5V or 3.3V) into integer values between 0 and 1023. On an Arduino UNO, for example, this yields a resolution between readings of: 5 volts / 1024 units or, 0.0049 volts (4.9 mV) per unit. See the table below for the usable pins, operating voltage and maximum resolution for some Arduino boards.

Q.17 Explain about integrated development environment (IDE).

Integrated development environments are designed to maximize programmer productivity by providing tight-knit components with similar user interfaces. IDEs present a single program in which all development is done. This program typically provides many features for authoring, modifying, compiling, deploying and debugging software. This contrasts with software development using unrelated tools, such as vi, GDB, GCC, or make.

One aim of the IDE is to reduce the configuration necessary to piece together multiple development utilities. Instead, it provides the same set of capabilities as one cohesive unit. Reducing setup time can increase developer productivity, especially in cases where learning to use the IDE is faster than manually integrating and learning all of the individual tools. Tighter integration of all development tasks has the potential to improve overall productivity beyond just helping with setup tasks. For example, code can be continuously parsed while it is being edited, providing instant feedback when syntax errors are introduced, thus allowing developers to debug code much faster and more easily with an IDE.

Some IDEs are dedicated to a specific programming language, allowing a feature set that most closely matches the programming paradigms of the language. However, there are many multiple-language IDEs.

Q.18 Explain the break and continue statement and its applications?

Break and continue statements

The break statement terminates a `while` or `for` loop completely. The continue statement terminates execution of the statements within a `while` or `for` loop and continues the loop in the next iteration. The following two examples demonstrate how these statements are used.

Note

Both these statements violate the rules of structured programming but are nonetheless widely used in special cases.

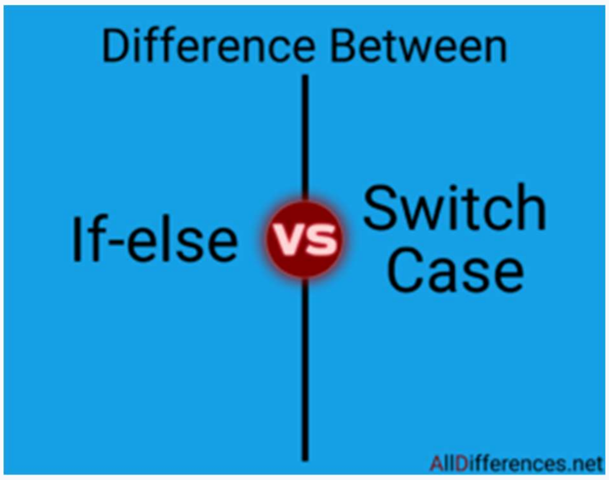
```
function break4error(parmArray, x)
{
    var i = 0;
    var n = 0;
    while (i < 10)
    {
       if (isNaN(parmArray[x]))
    {
       print(parmArray[i].toString + “ is not numeric. Please repair
       and run again”);
       break;
    }
       n = n + parmArray[i];
       i++;
    }
    return n;
}
```

Modifying the above example, the continue statement would work as follows:

```
function break4error(parmArray, x)
{
    var i = 0;
    var n = 0;
    while (i < 10)
    {
       if (isNaN(parmArray[x]))
       {
           print(parmArray[i].toString + “ is not numeric.
Continuing with
           next number”);
           i++;
           continue;
       }
           n = n + parmArray[i];
           i++;
       }
       return n;}
```

Q.19 Differentiate between else if and switch statements with examples.

"If-else" and "switch" are conditional statements. The key difference is that switch despatches *immediately* to the case concerned, typically via an indexed jump, rather than having to evaluate all the conditions that would be required in an if-elsechain, which means that code at the end of the chain is reached more slowly than code at the beginning.



Comparison Chart

IF-ELSE	SWITCH
If statement is used to select among two alternatives	The switch statement is used to select among multiple alternatives.
If can have values based on constraints.	Switch can have values based on user choice.
If implements Linear search.	Switch implements Binary search.
Float, double, char, int and other data types can be used in if condition.	Only int and char data types can be used in switch block.
It is difficult to edit the if-else statement, if the nested if-else statement is used.	It is easy to edit switch cases as, they are recognized easily.

Q.20 Draw and explain block diagram of switch case statement in 'C' programming.

Switch statement in C tests the value of a variable and compares it with multiple cases. Once the case match is found, a block of statements associated with that particular case is executed.

Each case in a block of a switch has a different name/number which is referred to as an identifier. The value provided by the user is compared with all the cases inside the switch block until the match is found.

If a case match is NOT found, then the default statement is executed, and the control goes out of the switch block. `switch( expression )`

```
{
        case value-1:
                        Block-1;
                        Break;
        case value-2:
                        Block-2;
                        Break;
        case value-n:
                        Block-n;
                        Break;
        default:
                        Block-1;
                        Break;
}
Statement-x;
```

https://www.guru99.com/images/1/020819_0506_SwitchCaseS1.png

Q.21 Compare : for, while or Do While?

do while loop is similar to while loop with the only difference that it checks for the condition after executing the statements, and therefore is an example of **Exit Control Loop.** https://media.geeksforgeeks.org/wp-content/uploads/loop3.png

while	do-while
Condition is checked first then statement(s) is executed.	Statement(s) is executed atleast once, thereafter condition is checked.
It might occur statement(s) is executed zero times, If condition is false.	At least once the statement(s) is executed.
No semicolon at the end of while. while(condition)	Semicolon at the end of while. while(condition);
If there is a single statement, brackets are not required.	Brackets are always required.

while	do-while
Variable in condition is initialized before the execution of loop.	variable may be initialized before or within the loop.
while loop is entry controlled loop.	do-while loop is exit controlled loop.
while(condition) { statement(s); }	do { statement(s); } while(condition);

Q.22 Write a code in Arduino to detect obstacle using Ultrasonic Sensor as well as IR sensor.

```
int const trigPin = 10;
int const echoPin = 9;
int const buzzPin = 2;
void setup()
{
pinMode(trigPin, OUTPUT); // trig pin will have pulses output
pinMode(echoPin, INPUT); // echo pin should be input to get pulse width
pinMode(buzzPin, OUTPUT); // buzz pin is output to control buzzering
}
void loop()
{
// Duration will be the input pulse width and distance will be the dista
nce to the obstacle in centimeters
int duration, distance;
// Output pulse with 1ms width on trigPin
digitalWrite(trigPin, HIGH);
delay(1);
digitalWrite(trigPin, LOW);
// Measure the pulse input in echo pin
duration = pulseIn(echoPin, HIGH);
// Distance is half the duration devided by 29.1 (from datasheet)
distance = (duration/2) / 29.1;
// if distance less than 0.5 meter and more than 0 (0 or less means over
range)
if (distance <= 50 && distance >= 0) {
// Buzz
digitalWrite(buzzPin, HIGH);
} else {
// Don't buzz
digitalWrite(buzzPin, LOW);
}
// Waiting 60 ms won't hurt any one
delay(60);
}
```

Ir

```
int analogInPin = A0;  // Analog input pin that the potentiometer is
attached to

int led =10;
int sensorValue = 0;        // value read from the pot
```

```
void setup() {
  // initialize serial communications at 9600 bps:
  Serial.begin(9600);
  pinMode(led, OUTPUT);
}

void loop() {
  // read the analog in value:
  sensorValue = analogRead(analogInPin);
  Serial.print("sensor = " );
  Serial.println(sensorValue);

  delay(200);

  if(sensorValue<80)
  {
    digitalWrite(led,HIGH);
  }
  else
  {
    digitalWrite(led,LOW);
  }
}
```

Q.23Write a code in Arduino to rotate a dc motor clockwise for 10 sec and anticlockwise for 35 sec.

```
/*
 * Created by ArduinoGetStarted.com
 *
 * This example code is in the public domain
 *
 * Tutorial page: https://arduinogetstarted.com/tutorials/arduino-dc-motor
 */

// constants won't change
const int ENA_PIN = 7; // the Arduino pin connected to the EN1 pin L298N
const int IN1_PIN = 6; // the Arduino pin connected to the IN1 pin L298N
const int IN2_PIN = 5; // the Arduino pin connected to the IN2 pin L298N

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pins as outputs.
  pinMode(ENA_PIN, OUTPUT);
  pinMode(IN1_PIN, OUTPUT);
  pinMode(IN2_PIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(IN1_PIN, HIGH); // control motor A spins clockwise
  digitalWrite(IN2_PIN, LOW);  // control motor A spins clockwise

  for (int speed = 0; speed <= 255; speed++) {
    analogWrite(ENA_PIN, speed); // control the speed
    delay(10);
  }

  delay(1000); // rotate at maximum speed 1 seconds in in clockwise direction

  // change direction
  digitalWrite(IN1_PIN, LOW);   // control motor A spins anti-clockwise
  digitalWrite(IN2_PIN, HIGH);  // control motor A spins anti-clockwise

  delay(1000); // rotate at maximum speed 1 seconds in in anti-clockwise direction

  for (int speed = 255; speed >= 0; speed--) {
    analogWrite(ENA_PIN, speed); // control the speed
    delay(10);
  }

  delay(1000); // stop motor 1 second
}
```

Q.24 What is serial communication? Discuss the different transmission modes in serial communication.
Que 5
Q.25 Write a program design traffic light control Signal one way.

```
int ledred=4;




int ledgreen=8;
int ledyellow=7;

void setup() {

  pinMode(ledred,OUTPUT);

  pinMode(ledgreen,OUTPUT);

  pinMode(ledyellow,OUTPUT);

  // put your setup code here, to run once:

}



void loop() {

  digitalWrite(ledred,HIGH);

  delay(10000);
```

```
digitalWrite(ledyellow,HIGH);

digitalWrite(ledred,LOW);

delay(1000);

digitalWrite(ledgreen,HIGH);

digitalWrite(ledred,LOW);

digitalWrite(ledyellow,LOW);

delay(10000);

digitalWrite(ledyellow,HIGH);

digitalWrite(ledgreen,LOW);

delay(2000);
```

  digitalWrite(ledyellow,LOW); } // put your main code here, to run repeatedly: